Analysis of Multiplexed Parse Trees for Almost Instantaneous VF Codes

Satoshi Yoshida* and Takuya Kida*

*Hokkaido University Graduate school of Information Science and Technology Division of Computer Science
### VF Code

**Compressed Data**

**Search/mining Directly**

**Algorithm on Compressed Data**

<table>
<thead>
<tr>
<th>Compressed Text</th>
<th>Input Text</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Fixed Length</td>
<td><strong>FF Code</strong> (Fixed length to Fixed length code)</td>
<td></td>
</tr>
<tr>
<td>Variable Length</td>
<td><strong>FV Code</strong> (Fixed length to Variable length code)</td>
<td></td>
</tr>
<tr>
<td>Tunstall Code</td>
<td><strong>VF Code</strong> (Variable length to Fixed length code)</td>
<td></td>
</tr>
<tr>
<td>Tunstall Code</td>
<td><strong>VV Code</strong> (Variable length to Variable length code)</td>
<td></td>
</tr>
</tbody>
</table>
Almost Instantaneous VF Code

YY code: AIVF code using multiple parse trees [Yamamoto and Yokoo, 2001]

AIVF code [Yamamoto and Yokoo, 2001]

utilizing context between blocks

utilizing unused codewords

Tunstall code [Tunstall, 1967]

Improves compression ratio considerably
A Problem of YY Code

Multiple parse trees

relatively to the number of kind of characters ($k$) in the input text

VMA tree [Yoshida & Kida, 2010]

Huge time and memory

$k$-1 parse trees

0010110
1001001
0001...
Research Goal

- We give a theoretical analysis of improvement on memory usage by multiplexing parse trees. [Yoshida & Kida, IEEE DCC 2010]
- We want to find a lower bound of the number of nodes in a VMA tree.

<table>
<thead>
<tr>
<th></th>
<th>Number of nodes in VMA tree</th>
<th>Number of nodes reduced by multiplexing</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Upper bound</strong></td>
<td>$V \leq Mk - \frac{1}{2}k^2 + \frac{1}{2}k - M + 1$</td>
<td>$\Omega(k^2)$ [YK DCC ‘10]</td>
</tr>
<tr>
<td></td>
<td>$= O(Mk - k^2)$ [YK DCC ‘10]</td>
<td></td>
</tr>
<tr>
<td><strong>Lower bound</strong></td>
<td>?</td>
<td>$D \geq \frac{1}{2}k^2 + \frac{1}{2}k - 3 = \Omega(k^2)$ [YK DCC ‘10]</td>
</tr>
</tbody>
</table>

Main Results

We found upper and lower bounds of the number of nodes in a VMA tree.

<table>
<thead>
<tr>
<th></th>
<th>Number of nodes in VMA tree</th>
<th>Number of nodes reduced by multiplexing</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Upper bound</strong></td>
<td>$V \leq Mk - \frac{1}{2}k^2 + \frac{1}{2}k - M + 1$</td>
<td>$D \leq \frac{1}{2}(k+1)(k+2) \frac{k^{\lceil \log_k M \rceil+2} - 1}{k-1} + k - 2$</td>
</tr>
<tr>
<td></td>
<td>$= O(Mk - k^2)$ [YK DCC '10]</td>
<td>$\leq O(Mk^3)$ [This paper]</td>
</tr>
<tr>
<td><strong>Lower bound</strong></td>
<td>$V \geq \frac{1}{k-1}k^{\lceil \log_k M \rceil+1} \geq \frac{M}{k-1}$</td>
<td>$D \geq \frac{1}{2}k^2 + \frac{1}{2}k - 3 = \Omega(k^2)$</td>
</tr>
<tr>
<td></td>
<td>$= \Omega \left( \frac{M}{k} \right)$</td>
<td>[YK DCC '10]</td>
</tr>
<tr>
<td></td>
<td>[This paper]</td>
<td></td>
</tr>
</tbody>
</table>

$k$: number of kind of characters (alphabet size)

$M$: number of codewords
Let $\Sigma$ be a finite alphabet.

Elements in alphabet are sorted in descending order of their probabilities.

We assume information source is memoryless.

We simply say “encoding” encoding to the sequence on $\{0, 1\}$.

\[ \text{aabbbbc} \quad \xrightarrow{\Sigma = \{a, b, c\}} \]
input text: aabbbbc

Each branch is labeled by a symbol in $\Sigma=\{a, b, c\}$.

Incomplete internal node
An internal node which doesn’t have $k$ children.

Input text is parsed into blocks and we output codeword corresponding to the block.

compressed data sequence: 001 101 101 101 101 111

Each leaf node and incomplete internal node has a codeword.
An Idea of YY Code [Yamamoto and Yokoo 2001]

We switch multiple parse trees according to the context.
Basic Idea of VMA tree
[Yoshida & Kida 2010]

- Problem: time and space consuming
  - We construct $k - 1$ parse trees.
- We can reduce total number of nodes by sharing nodes.
  - We need to mark each node $n$ in a VMA tree in order to tell which trees the node belongs to.
  - We can realize that by holding the least $i$ such that $n$ belongs to $T_i$. 

\[ T_0 \quad T_1 \quad T_2 \quad \cdots \quad T_{k-2} \]

multiple parse trees \quad \quad \quad \quad \quad \quad \quad \quad VMA tree
Where Can We Share?

From this theorem, we can tell which trees a node belongs to easily.

- **Theorem**
  Let $S_j^{(i)}$ be the subtree of $T_i$ that consists of all the nodes under the node corresponding with $a_j$, which is direct child of the root. Then $S_{i+j}^{(i+1)}$ completely covers $S_{i+j}^{(i)}$. We denote this relation by $S_{i+j}^{(i)} < S_{i+j}^{(i+1)}$.

![Diagram showing tree structures and subtree relationships](image-url)
From the theorem, we have:

\[ S_1^{(0)} = S_1, \]
\[ S_2^{(0)} < S_2^{(1)} = S_2, \]
\[ S_3^{(0)} < S_3^{(1)} < S_3^{(2)} = S_3, \]
\[ \vdots \]
\[ S_{k-2}^{(0)} < \ldots < S_{k-2}^{(k-3)} = S_{k-2}, \]
\[ S_{k-1}^{(0)} < \ldots < S_{k-1}^{(k-2)} = S_{k-1}, \]
\[ S_k^{(0)} < \ldots < S_k^{(k-2)} = S_k. \]
How Many Nodes in a VMA Tree?

- Theorem

  The number of nodes in a VMA tree is not less than \( \frac{1}{k - 1} k^{\lceil \log_k M \rceil + 1} \).
Proof Sketch

Largest subtree and it remains in VMA tree.

The VMA tree is smallest when \( \Pr(a_1) = \Pr(a_2) = \cdots = \Pr(a_k) \).
Proof Sketch

# of codewords:

\[
\begin{align*}
S_1^{(0)} & : \frac{M}{k} \\
S_2^{(1)} & : \frac{M}{k-1} \\
\vdots & \vdots \\
S_{k-2}^{(k-3)} & : \frac{M}{3} \\
S_{k-1}^{(k-2)} & : \frac{M}{2} \\
S_k^{(k-2)} & : \frac{M}{2}
\end{align*}
\]

# of nodes:

\[
\begin{align*}
\frac{k|\log_{k-1}^k M|+1 - 1}{k - 1} & \quad \frac{k|\log_{k-2}^k M|+1 - 1}{k - 1} & \quad \frac{k|\log_{k-3}^k M|+1 - 1}{k - 1} & \quad \frac{k|\log_{k-4}^k M|+1 - 1}{k - 1} & \quad \frac{k|\log_{k-5}^k M|+1 - 1}{k - 1}
\end{align*}
\]

Summation is not less than \( \frac{1}{k-1} k|\log_k M|+1 \).
Experiments

- **Comparison**
  - Total number of nodes and upper bound and lower bound of nodes in VMA tree on random sequence.

- **Algorithms**
  - YY coding (YY)
  - Encoding using a VMA tree (VMA)

- **Environments**
  - **CPU:** Intel Pentium® 4 Processor 3.0GHz Hyper Threading
  - **Memory:** 2GB
  - **OS:** Debian GNU/Linux 5.0
  - **Language:** C++
  - **Compiler:** g++4.3

- **Codeword length:** 12 bits
Exp: Comparison in the numbers of nodes

the number of nodes on uniform distribution

the number of nodes on Zipf distribution

\[ (10^3) \]
Conclusions

- We found an upper bound and a lower bound of the number of nodes in a VMA tree.

<table>
<thead>
<tr>
<th></th>
<th>Number of nodes in VMA tree</th>
<th>Number of nodes reduced by integration</th>
</tr>
</thead>
<tbody>
<tr>
<td>Upper bound</td>
<td>( Mk - \frac{1}{2}k^2 + \frac{1}{2}k - M + 1 )</td>
<td>( \frac{1}{2}(k + 1)(k + 2) \frac{k^{\log_k M} + 2 - 1}{k - 1} + k - 2 )</td>
</tr>
<tr>
<td>Lower bound</td>
<td>( \frac{1}{k - 1} k^{\log_k M} + 1 )</td>
<td>( \frac{1}{2}k^2 + \frac{1}{2}k - 3 )</td>
</tr>
</tbody>
</table>

Future work

- Finding reduction ratio of the number of nodes.
- Finding worst input.